Hindawi
Complexity
Volume 2020, Article ID 9428612, 23 pages
https://doi.org/10.1155/2020/9428612

Research Article

Deep Ensemble Learning for Human Action Recognition in Still Images

Xiangchun Yu,[1,2] **Zhe Zhang**,[2] **Lei Wu**,[2] **Wei Pang**,[3] **Hechang Chen**,[2,4] **Zhezhou Yu**,[2] **and Bin Li**[5]

[1]*School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China*
[2]*College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[3]*School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh EH14 4AS, UK*
[4]*School of Artificial Intelligence, Jilin University, Changchun 130012, China*
[5]*School of Information Engineering, Northeast Electric Power University, Jilin 132012, China*

Correspondence should be addressed to Xiangchun Yu; yuxc@jxust.edu.cn

Received 10 November 2019; Accepted 2 January 2020; Published 31 January 2020

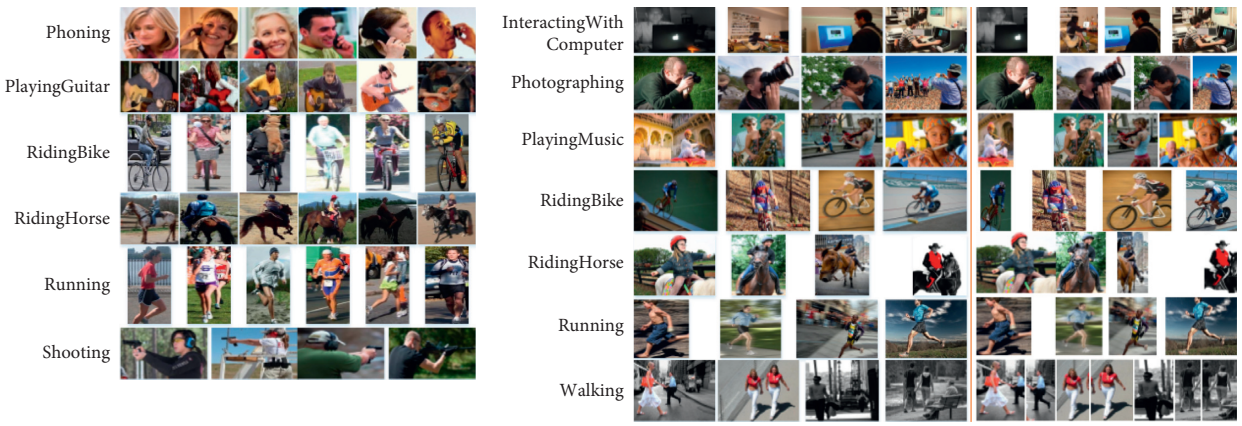
Academic Editor: Sarangapani Jagannathan

Numerous human actions such as "Phoning," "PlayingGuitar," and "RidingHorse" can be inferred by static cue-based approaches even if their motions in video are available considering one single still image may already sufficiently explain a particular action. In this research, we investigate human action recognition in still images and utilize deep ensemble learning to automatically decompose the body pose and perceive its background information. Firstly, we construct an end-to-end NCNN-based model by attaching the nonsequential convolutional neural network (NCNN) module to the top of the pretrained model. The nonsequential network topology of NCNN can separately learn the spatial- and channel-wise features with parallel branches, which helps improve the model performance. Subsequently, in order to further exploit the advantage of the nonsequential topology, we propose an end-to-end deep ensemble learning based on the weight optimization (DELWO) model. It contributes to fusing the deep information derived from multiple models automatically from the data. Finally, we design the deep ensemble learning based on voting strategy (DELVS) model to pool together multiple deep models with weighted coefficients to obtain a better prediction. More importantly, the model complexity can be reduced by lessening the number of trainable parameters, thereby effectively mitigating overfitting issues of the model in small datasets to some extent. We conduct experiments in Li's action dataset, uncropped and 1.5x cropped Willow action datasets, and the results have validated the effectiveness and robustness of our proposed models in terms of mitigating overfitting issues in small datasets. Finally, we open source our code for the model in GitHub (https://github.com/yxchspring/deep_ensemble_learning) in order to share our model with the community.

1. Introduction

Human action recognition [1–6] is one of the most important research fields in computer vision. Although recognizing the motion of human action in video can provide discriminative clues for classifying one specific action, many human actions (e.g., "Phoning," "InteractingWithComputer," and "Shooting," as shown in Figure 1), can be represented by one single still image [2]. In particular, certain actions (e.g., "PlayingGuitar," "RidingHorse," and "Running," as shown in Figure 1) may require static cue-based approaches even if those motions in videos are available [2]. To recognize these human actions with video-based approaches mentioned above [5, 6, 8] may be inappropriate due to their slight action changes without distinguishability. Its static features by nature motivate us to address those human action recognition tasks in still images [2]. Classifying human actions in still images is a more challenging task, especially when only one single image is available along with disturbance and cluttered background.

More and more work [9–14] has recently focused on human action recognition in still images. In this research, we strive to investigate a robust human action model for still images which does not need manual feature engineering,



(a) (b)

Figure 1: Example images for (a) Li's action [7] and (b) Willow action datasets [2]. In (b), the left part shows the original images from Willow action dataset, and the right part represents the 1.5x cropped images of annotated bounding boxes of human actions.

explicit body pose estimation and reasoning, or part-based representations. In this research, we concentrate on employing the deep ensemble learning to address such tasks. First of all, we explore the application of nonsequential network topology in human action recognition in still images. Specifically, we propose to attach a nonsequential convolutional neural network (NCNN) module to the pretrained model. The NCNN module has three independent branches, and each branch can learn the spatial- and channel-wise features separately. Then, the end-to-end NCNN-based model is trained to learn the domain-specific knowledge in small datasets. Secondly, different kinds of models may discover multiple aspects of the "truth," so we further examine the benefits of deep ensemble learning in terms of improving classification performance. We propose an end-to-end deep ensemble learning based on the weight optimization (DELWO) model to fuse the information derived from multiple deep models to achieve better performance. DELWO also has a nonsequential network topology and is a generalized multi-input model with each pretrained model as an input. Besides, we also propose a deep ensemble learning based on voting strategy (DELVS) model to integrate prediction results using different voting strategies to obtain better predictions. Our proposed models can side-step the trivial tasks related to manual feature design, body part-based modeling, and action poselet-based representation, etc.

In practice, how to mitigate overfitting issues is one important concern in computer vision tasks when only a few data are available for training. For instance, in Li's action dataset [7] as shown in Figure 1, only 180 images are used as the training set. For another example, only 208 images and 280 images are utilized as the training set for uncropped and 1.5x cropped Willow action dataset [2], respectively. We are committed to construct the deep CNN models which can mitigate the overfitting issue in small datasets to some degree. The main features of our proposed models in this research are as follows:

(1) Our proposed NCNN-based model and DELWO model are both end-to-end, which can directly produce a prediction for one input and also make the batch processing operation possible for model training. It can greatly reduce the memory consumption and make it feasible to train our own deep learning models on a single PC with CPU.

(2) Our NCNN-based model and DELWO model have a nonsequential network topology. The advantages of the NCNN-based model are as follows: first, it can automatically learn information from different channels; second, it can contribute to fine-tuning the top layers by optimizing weight parameters of the NCNN module so that the model can be more competent for a domain-specific task. DELWO model fuses the deep information derived from multiple models and then automatically exploits the advantages of each model from the data.

(3) We propose the deep ensemble models, DELWO and DELVS. DELWO can avoid manually specifying weight coefficients of various models. DELVS model needs to determine weight parameters in advance, and then multiple models are combined to pool together the prediction using different voting strategies and endeavor to achieve better performance. These two kinds of deep ensemble models are proposed to explore their performance for a domain-specific task.

The whole framework of our proposed algorithm is elaborated in Figure 2.

The rest of the paper is organized as follows. We first review the related work for human action recognition in still images in Section 2. In Section 3, we elucidate the specific methodology including data processing and model construction. We report the experimental results in Section 4, and this is followed by the conclusions and future work drawn in Section 5.

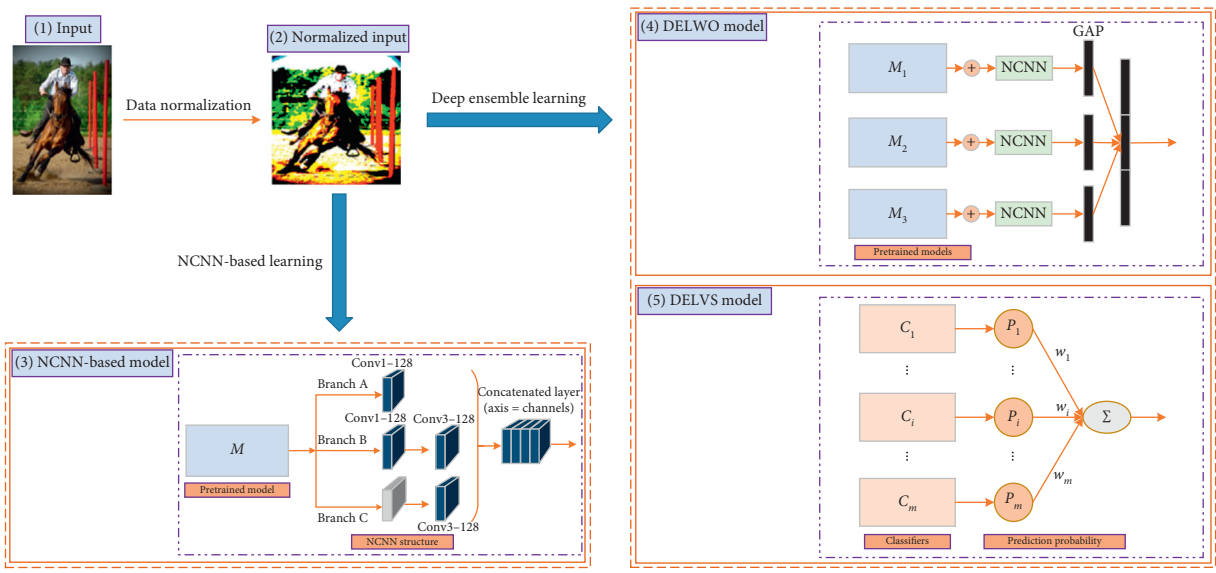


Figure 2: The whole framework of our proposed algorithm. Firstly, the original input (1) is converted into the normalized input (2). Then, our proposed NCNN module is attached to the top of the pretrained model (e.g., VGG16). Furthermore, the NCNN-based model is trained to carry out the NCNN-based learning (3). In order to take advantage of ensemble learning, the DELWO model (4) aims to fuse multiple different types of models to directly explore more aspects of the "truth," and finally, the DELVS model (5) integrates multiple classifiers to obtain the final prediction using different voting strategies.

2. Related Work

Existing work mainly focuses on feature engineering (e.g., bag-of-features), body part-based modeling, or action poselet-based representation, etc. for human action recognition. Delaitre et al. [2] studied human action recognition in still images using the bag-of-features model [2]. Qi et al. [14] proposed to construct a hint-enhanced CNN framework by jointly learning the pose hints and deep feature extraction. Kong et al. [15] proposed to extract depth motion maps pyramid descriptor for each action followed by the classifier of discriminative collaborative representation to perform human action recognition. Gupta et al. [16] utilized the body pose as a clue for action recognition. Zhang et al. [17] proposed a foreground trajectory extraction approach based on a saliency sampling strategy intending to lessen the reduction of valid trajectories of action. Felzenszwalb et al. [18] proposed a structural part-based model to represent human actions. Ko et al. [19] proposed an action poselet-based approach and a two-layer classification model to infer human actions. Cai et al. [20] proposed an improved CNN for conducting human action recognition by extracting depth sequence features using depth motion maps as well as obtaining the three projected maps: the front, side, and top views.

With the outstanding performance of deep learning in computer vision, it will be one important step forward to construct a deep learning model for automatically analyzing the body pose and perceive its background information. However, training a deep CNN model from scratch using a small dataset often suffers from overfitting issues. Data augmentation, a powerful technique to mitigate overfitting, can generate more training data using random transformations such as rotation, shift, and shear. It can make our model never see the same sample twice during training and therefore enable the exposure of our model to more aspects of the data. Another technique is to adopt pretrained deep networks (e.g., VGG16 [21]) as the initial model to extract deep features, which makes our deep learning model effective even when only a small dataset is available.

The traditional CNN models (e.g., LetNet-5 [22] and VGG16 [21]) are sequential ones in the sense that they have linear stacks of layers. These sequential models may be inflexible in some cases. The inception module proposed by Szegedy et al. [23] possesses a nonsequential network topology: the model follows the structure of a directed acyclic graph. Its input in the inception module is separately processed by certain parallel branches followed by a concatenated layer which merges back the output of each branch into one single tensor. The number of trainable parameters of networks considerably determines the degree of model overfitting. Lin et al. [24] proposed the global average mapping (GAP) to replace the fully connected layer followed by the Softmax layers in CNN. GAP greatly reduces the number of trainable parameters and make the model lighter and thus alleviate overfitting. Multi-input models [25–28] is another kind of nonsequential network topology, and it has multiple input layers which can make full use of multimodal or multiple types of data. Nickfarjam and Ebrahimpour-Komleh [25] adopt the deep belief networks with multi-input topology to conduct shape-based human action classification and improved model performance.

In addition, ensemble learning [29, 30] can pool together different models to achieve better performance. It pools different classifiers and incorporates their predictions either by weighted coefficients or majority votes. It can take

advantage of different models to explore as many parts of the "truth" of the data as possible [31].

3. Methodology

3.1. Data Processing. The availability of very few data is a common situation when a classification model needs to be trained for image recognition. In order to mitigate the overfitting issues, we adopt the data augmentation technique to improve the performance of CNN. Data augmentation can generate more training data via random transformations of training images and enable to expose the training model to more possible aspects of the data distribution [31]. The random transformations in this research contain rotation within 0–90 degree, width shift within 0–0.2, height shift within 0–0.2, shear within 0–0.2, zoom within 0–0.2, horizontal flip, and vertical flip. Besides, we conduct the image-wise centralization to implement the sample normalization.

3.2. Nonsequential Convolutional Neural Network Model. In this section, we present our proposed NCNN module. First, our proposed NCNN-based model is applicable to small datasets. Second, it is an end-to-end model that directly produces the output for each input sample. More importantly, it can well contribute to reducing memory consumption by batch processing and make it possible to train our models when only CPUs are available (although GPUs are better). Compared to adding a convolution layer of the same number of filters, it makes NCNN-based model lighter and improves the generalization ability of model.

Figure 3 shows the structure of the VGG16 in this research, and the VGG16_base module and Classifier module are connected by a GAP layer. The VGG16_base module is based on VGG16 [21], and the weights of convolution layers are initialized from the pretrained VGG16, and those layers are frozen when carrying out model training.

In order to mitigate overfitting, speed up model training, and improve generalization ability, we construct the NCNN-based model (e.g., VGG16_NCNN). Specifically, the VGG16_NCNN incorporates three modules, the VGG16_base module, NCNN module, and the classifier module after the GAP layer. The whole structure of VGG16_NCNN is illustrated in Figure 4. As shown in Figure 4, the NCNN module has three branches. Branch A possesses one "Conv1–128" layer, which means the kernel size is 1×1, and the filter size is 128, where K denotes the kernel size and F denotes the filter size for "ConvK-F." Similarly, Branch B has two convolution layers: one "Conv1–128" layer followed by one "Conv3–128" layer. Branch C has one 3×3 average pooling layer followed by a "Conv1–128" layer. Finally, the last activation output of each branch is concatenated together to form the resulting concatenated layer. Same as the VGG16, a three-layer classifier module is constructed after the GAP layer.

The weights of the VGG16_base module are initialized from the pretrained VGG16 model, and the ones of the NCNN module are randomly initialized. When we conduct model training, the weights of the VGG16_base module are frozen and the purpose is to prevent the backpropagated error via the randomly initialized layers from destroying the pretrained convolution layers.

The advantage of having NCNN module and a GAP layer is that it can effectively decrease the trainable parameters. Therefore, our model will be lighter, which can greatly facilitate the mitigation of model overfitting and promote generalization ability. Specifically, when the size of input samples is 224×224, the number of parameters between the last layer of the VGG16_base module and the first layer of classifier module will be $(7 \times 7 \times 512) \times 2048 + 2048 = 51,382,272$. For VGG16, the number of parameters between the GAP layer and the first layer of classifier module will be $512 \times 2048 + 2048 = 1,050,624$. For VGG16_NCNN, the number of parameters between the GAP layer and the first layer of classifier module will be $384 \times 2048 + 2048 = 788,480$.

Without the NCNN module and a GAP layer, the total number of trainable parameters will be 70,307,655. However, the total number of trainable parameters of the VGG16 (see Figure 4) is 5,261,319, and that of our proposed VGG16_NCNN model is 5,868,039. Compared with VGG16, although the total number of trainable parameters of VGG16_NCNN is slightly larger, the effectiveness of the VGG16_NCNN is enhanced. In other words, by training the parameters of the NCNN module, we can effectively fine-tune the model (i.e., fine-tune layers in the red dotted box of Figure 4), making the model more suitable for our domain-specific tasks. The number of trainable parameters of a model determines the complexity of the model. A reasonable situation is that the size of the data and the complexity of the model can be effectively matched. Therefore, by adding the NCNN module and a GAP layer to the model, the overfitting issues can be mitigated to some extent.

To train our model, we need to minimize the following loss function of categorical crossentropy:

$$\begin{aligned} L &= -\frac{1}{N}\sum_{i=0}^{N-1} Y_i \log P_i \\ &= -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=0}^{K-1} y_{i,k} \log p_{i,k}, \end{aligned} \tag{1}$$

where $p_{i,k}$ denotes the probability of predicting the sample i to class k, N denotes the sample size, and $y_{i,k}$ is the true label for sample i belonging to class k.

Specifically, when the first "FC-2048" of the classifier module is regarded as the input, formula (1) can be further expressed as

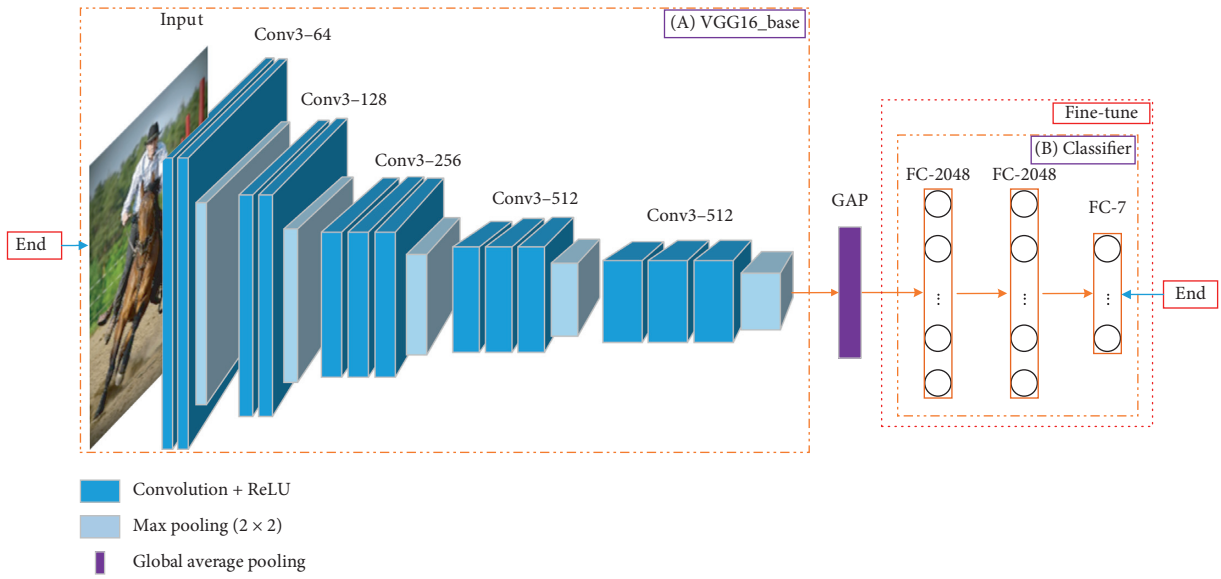


Figure 3: The specific structure of VGG16 [21]. The convolution layers module (i.e., VGG16_base) based on VGG16 and the classifier module are connected by the GAP layer.

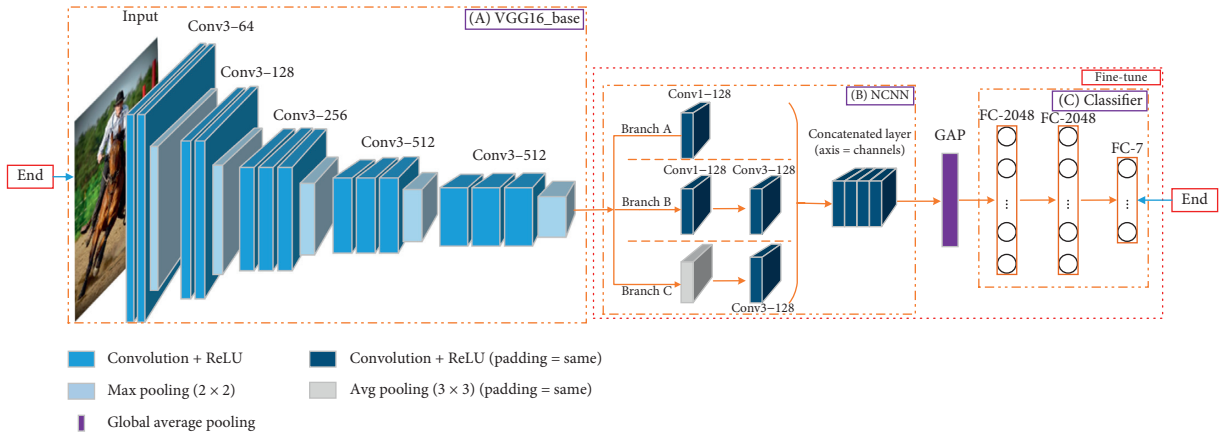


Figure 4: The specific structure of VGG16_NCNN. It incorporates three modules: the VGG16_base module, NCNN module followed by the GAP layer, and the classifier module.

$$
\begin{aligned}
L &= -\frac{1}{N}\sum_{i=0}^{N-1} Y_i \log P_i \\
&= -\frac{1}{N}\sum_{i=0}^{N-1} Y_i \log\left(\sigma\left(\left(W^{(\mathrm{FC2})}\right)^T\right)h\left(\left(W^{(\mathrm{FC1})}\right)^T\right)X\right) \\
&= -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=0}^{K-1} y_{i,k}\log\left(\sigma\left(\sum_{j=0}^{M-1} w_{kj}^{(\mathrm{FC2})} h\left(\sum_{d=0}^{D-1} w_{jd}^{(\mathrm{FC1})} x_d + w_{j0}^{FC1}\right) + w_{k0}^{(\mathrm{FC2})}\right)\right),
\end{aligned}
\tag{2}
$$

where "FC1" and "FC2" denote the first and second "FC-2048" layers of the classifier module, σ denotes the sigmoid function, h denotes the ReLU activation function, and D and M represent the numbers of nodes in "FC1" and "FC2," respectively.

When an unseen sample comes in, the following function is employed to produce a prediction class for sample i:

$$
\widehat{y}_i = \max_k\left(p_{i,k}\right). \tag{3}
$$

3.3. Deep Ensemble Learning Based on Weight Optimization. Different deep models will focus on different aspects of the "truth." Therefore, in order to better incorporate more information about the "truth," we design an end-to-end DELWO model that can directly produce the output for each input sample. As shown in Figure 5, the training data are fed into multiple deep models with the NCNN module and then we conduct the GAP concatenation from each model to form a longer layer (i.e., GAP concatenation module) which is connected to the classifier module. In this research, we define 3 kinds of DELWO models:

(1) DELWO1 fuses VGG16 [21], VGG19 [21], and ResNet50 [32] in deep model module, and the filter size of GAP is set to 128 in GAP concatenation module

(2) DELWO2 fuses VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN in deep model module, and the filter size of GAP is set to 128 in GAP concatenation module

(2) DELWO3 fuses VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN in deep model module, and the filter size of GAP is set to 384 in GAP concatenation module

Figure 5 elaborates the specific process of human action classification using DELWO2. When the model is well trained, the testing data are fed into it to produce the final prediction. The specific training steps are similar to those of the NCNN-based model.

3.4. Deep Ensemble Learning Based on Voting Strategy. Ensemble learning is a powerful technique to achieve better prediction results. We assume that different models focus on different aspects of the "truth model." Therefore, pooling together different models can discover as many parts of "truth" as possible. In this research, we pool together multiple deep models using DELVS with an aim to obtain better prediction results. The prediction class for sample i is estimated using the following 3 kinds of functions.

3.4.1. Hard Voting. Hard voting strategy (i.e., majority voting) aims to predict the final class label via computing the label majority of all classifiers. And the function is shown in the following equation:

$$\widehat{y}_i = \text{mode}\left\{C_1(x_i), C_2(x_i), \ldots C_j(x_i), \ldots, C_m(x_i)\right\}, \quad (4)$$

where x_i denotes sample i, $C_j(x_i)$ denotes the prediction label of j-th classifier, mode function is used to compute the majority of all the prediction labels, and m denotes the number of classifiers.

3.4.2. Soft Voting. Soft voting strategy aims to predict the final class label via computing the sum of prediction probabilities of each class among all classifiers. The label is assigned to the category that gets the highest probability sum. And the function is shown in the following formula:

$$\widehat{y}_i = \arg\max_k \sum_{j=1}^{m} w_j p_{i,k}^j, \quad (5)$$

where w_j denotes the weight of the j-th classifier, $p_{i,k}^j$ denotes the prediction probability of the j-th classifier predicting the sample i into k-th class, and m denotes the number of classifiers. It is worth noting that the weight w_j is set to $1/m$ in this voting strategy.

3.4.3. Tuning Weight Voting. Soft voting adopts the weighted average strategy, and it sometimes does not highlight the differences in model contributions. Therefore, we adopt the grid search approach to search for the optimal weights to obtain better predictions. Specifically, the weight parameters are optimized by setting a step size to find the best overall accuracy within a specific weight parameter range. The weight coefficients corresponding to the best overall accuracy are the optimal results. And the function is shown in formula (6), which is similar to formula (5):

$$\widehat{y}_i = \arg\max_k \sum_{j=1}^{m} w_j' p_{i,k}^j, \quad (6)$$

where $\{w_j'\}, j = 1, \ldots, m$ is the optimal weight coefficients for each classifier and $\sum_{j=1}^{m} w_j' = 1$.

In this research, we define 3 kinds of DELVS models, which correspond to DELVS1, DELVS2, and DELVS3:

(1) DELVS1 integrates the predictions of VGG16 [21], VGG19 [21], and ResNet50 [32] using the three voting strategies mentioned above to obtain the final prediction results

(2) DELVS2 integrates the predictions of VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN using the three voting strategies mentioned above to obtain the final prediction results

(3) DELVS3 integrates the predictions of VGG16 [21], VGG19 [21], and ResNet50 [32] and VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN using the three voting strategies mentioned above to obtain the final prediction results

Figure 6 elaborates the specific process of human action classification using DELVS. When the testing data are fed into the m classification models, three kinds of voting strategies are utilized to obtain the final prediction, respectively.

4. Results

In this section, we evaluate the performance of our proposed models in the following datasets: Li's action dataset, Willow action dataset, and 1.5x cropped Willow action dataset. We first demonstrate the specific experimental setup and then present the detailed experimental results.

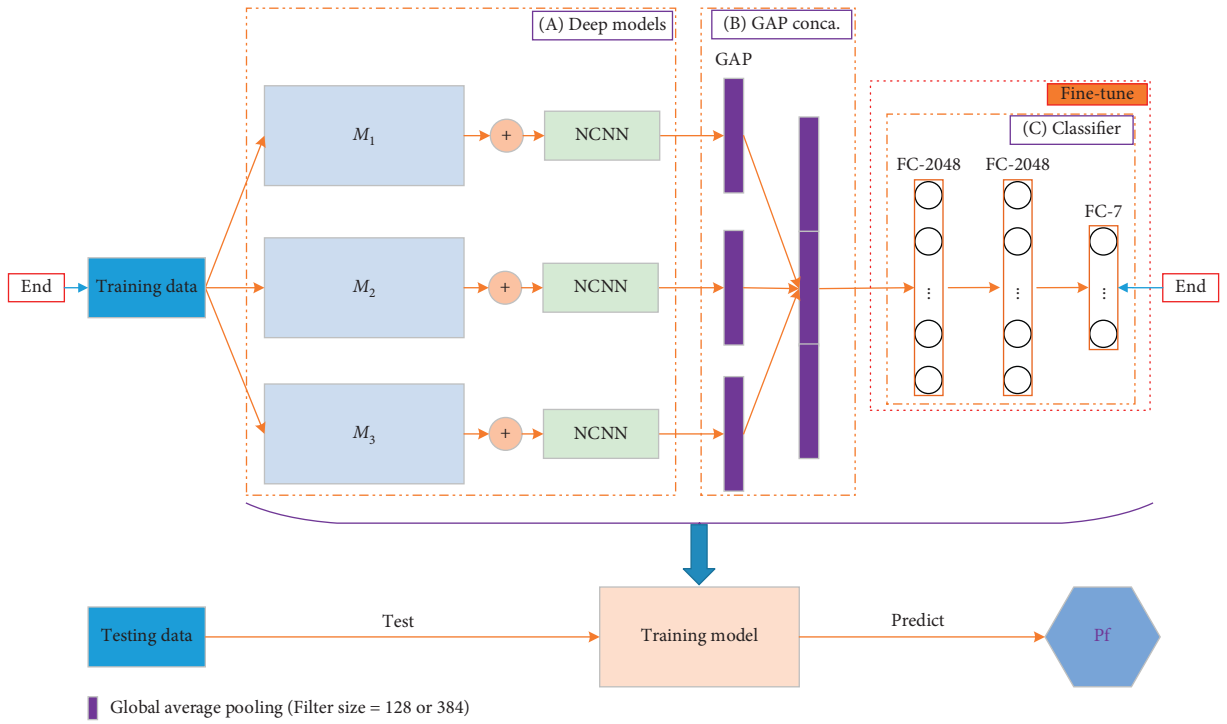


Figure 5: The whole framework for DELWO model. (1) For training phase, training data are fed into multiple deep models, then the GAP layers are concatenated into one long layer; finally, the fused information derived from multiple deep models is fed into the classifier module and conducts weight optimization. (2) For testing phase, testing data are fed into the training model to get the final prediction probability.

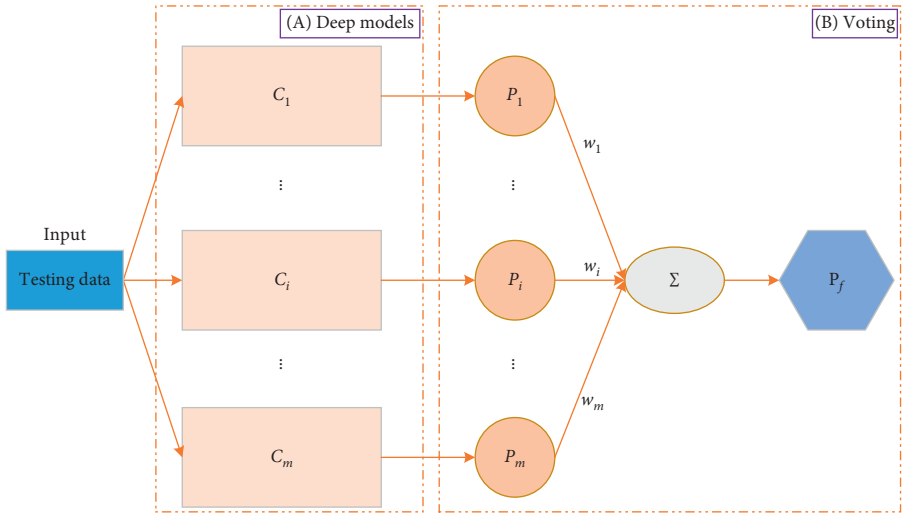


Figure 6: The specific process of DELVS model. Testing data are fed into the classification models, and three kinds of voting strategies including hard voting, soft voting, and tuning weight voting are utilized to obtain the final prediction, respectively.

4.1. Experimental Setup

4.1.1. Datasets. For Li's action dataset (the data and code of Li's paper are available at https://github.com/lipiji/PG_BOW_DEMO), six common human action categories are released with a total of 240 images, and these actions are "Phoning," "PlayingGuitar," "RidingBike," "RidingHorse," "Running," and "Shooting," respectively [7]. These images are well cropped in advance, and the images in each category are of the same size (see Figure 1(a)). We randomly choose 30 images for training, 10 images for validation, and the remaining 20 images for testing for each class.

For the Willow action dataset (the data and code of Delaitre's paper are available at https://www.di.ens.fr/willow/research/stillactions/), seven human action categories are collected by Delaitre et al. [2] with 911 images, and they are "InteractingWithComputer," "Photographing," "PlayingMusic," "RidingBike," "RidingHorse," "Running," and "Walking," respectively. This dataset contains more challenging noncropped consumer photographs, where

natural viewpoint variation, *occlusions*, scene layout, variations related to object appearance, and people's clothing are present among them [2]. In addition, the images in each category are of different sizes. The location of people in each image is manually annotated with bounding boxes in this dataset. In order to evaluate the performance of our model, we conduct the experiments on both uncropped (i.e., original images with background) Willow action datasets and 1.5x cropped (i.e., rescale the bounding box of human action to 1.5 times). The training, validation, and testing set settings are consistent with those in Delaitre's work [2].

4.1.2. Model Parameter Setup. For each model, we adopt the pretrained model to initialize the weights, and the weights of the NCNN module and the following classifier module are randomly initialized. It is worth noting that all the convolution layers of the pretrained model are frozen during training. It tries to make sure the weights of pretrained convolution layers will not be destroyed. Otherwise, the backpropagated error via the randomly initialized classifier layers will be too large, which makes our model more difficult to train. We conduct the experiments on multiple pretrained models including VGG16 [21], VGG19 [21], InceptionV3 [23], DenseNet [33], ResNet50 [32], and MobileNet [34]. However, only VGG16, VGG19, and ResNet50 achieve good performance in human behavior recognition. Therefore, we will carry out further research on these models. In order to assess the performance of our proposed models, the comparison algorithms including speeded-up robust features (SURF) [35], bag-of-features (BOF) [36], and pyramid bag-of-features (PBOF) [37] are used. In order to compare with nondeep ensemble learning approaches, we further conduct the comparison algorithms including bagging-based ensemble learning [38] (e.g., Random Forests (RF [39])), boosting-based ensemble learning [40] (e.g., gradient boosting machines (GBMs) [41]), and voting-based ensemble learning [42] among support vector machine (SVM), RF, and GBM classifiers. All the evaluation of nondeep ensemble learning approaches is based on the 512-dimensional GIST descriptors. At the same time, the GIST [43, 44] (with SVM classifier) method was also attached to the comparison experiments.

4.2. Experimental Results

4.2.1. Results for Nonsequential Convolutional Neural Network Model. Table 1 shows the classification performance in Li's action dataset. The SURF achieves the worst performance. The BOF and PBOF achieve better performance over SURF but still do not exceed the models we propose. For the nondeep ensemble learning approaches, only the performance of the voting-based approach surpasses that of GIST, while the remaining RF and GBM achieve worse results compared with the GIST approach. VGG16, VGG19, and ResNet50 perform well, and this reveals the pretrained weights are feasible for conducting human action categories in Li's action dataset. More importantly, VGG19_NCNN and ResNet50_NCNN still outperform the baseline models in terms of the overall accuracy and loss. Particularly, ResNet50_NCNN model achieves the best overall accuracy and least loss. From Table 1 and Figure 7, we are convinced that the NCNN-based model works the same or better over the baseline model despite a slightly higher loss are obtained for VGG16_NCNN.

Table 2 presents the results of using the NCNN-based model in the Willow action dataset. This is a more challenging dataset, the natural and challenging disturbances occurred in the images, including viewpoint variation, occlusions, scene layout, and variations related to object appearance and people's clothing. Therefore, the seven comparison algorithms have failed in classifying those human actions. In particular, the performance of nondeep ensemble learning approaches does not exceed the GIST approach. From Table 2 and Figure 8, we can see that NCNN-based model basically outperforms the baseline model in terms of the overall accuracy and loss except for the VGG19_NCNN in terms of overall accuracy.

Besides, we can see that all the models fail when classifying the "Photographing," "Running," and "Walking" actions. Because these three actions are similar (see Figure 1(b)), the uncertainty will arise when producing their predictions. As shown in Figure 8, for the "Photographing" action row, it was misclassified into "Walking" with a rate of 0.28 by the VGG16_NCNN, "Walking" with a rate of 0.38 by VGG19_NCNN, and "Walking" with a rate of 0.16 by the ResNet50_NCNN. For the "Running" action row, VGG16_NCNN and VGG19_NCNN achieve a better classification ability compared with the ResNet50_NCNN. This reveals that VGG16_NCNN and VGG19_NCNN can better discriminate "Running" and "Walking" to a certain extent. However, for the "Walking" action row, all the models cannot well tell the differences between this action and the "Running" action.

Table 3 presents the results of using the NCNN-based model in the 1.5x cropped Willow action dataset. From Table 3 and Figure 9, we can see that NCNN-based model outperforms the baseline model in terms of the overall accuracy and loss. Similar to the results in Table 2, the performance of nondeep ensemble learning approaches does not exceed the GIST approach. It is worth noting that VGG19 and VGG19_NCNN perform better when classifying the "Running" action compared with other models, and the ResNet50_NCNN performs better when classifying the "Walking" actions compared with the other models. ResNet50_NCNN achieves the best overall accuracy and loss among all models, and it well validates the effectiveness of our proposed NCNN-based method.

The whole experimental results of thirteen algorithms in those three datasets are shown in Figure 10. Figure 10 shows that almost all comparative methods including SURF, BOF, PBOF, GIST, and nondeep ensemble learning approaches including RF, GM, and voting fail in this task, and only BOF, PBOF, GIST, RF, GM, and voting approaches show good results in Li's action dataset. The performance of all the deep learning-based models is better than that of the comparison algorithms.

4.2.2. Results for Deep Ensemble Learning Based on Weight Optimization. Table 4 shows the results of using DELWO1,

Table 1: Results for NCNN in Li's action dataset.

Algorithm	Sensitivity for each class						Overall	
	Phoning	P.Guitar	R.Bike	R.Horse	Running	Shooting	Acc	Loss
SURF	0.45	0.1	0.2	0.2	0.05	0.2	0.2	NA
BOF	0.8	0.6	0.75	0.75	0.6	0.9	0.7333	NA
PBOF	0.95	0.75	0.8	0.8	0.75	0.95	0.8333	NA
GIST	0.70	0.75	0.65	0.85	0.70	0.70	0.725	NA
RF	0.70	0.75	0.55	0.75	0.85	0.70	0.7167	NA
GBM	0.55	0.55	0.75	0.70	0.65	0.60	0.6333	NA
Voting	0.85	0.75	0.80	0.85	0.75	0.70	0.7833	NA
VGG16	1	0.95	0.9	1	1	0.85	0.95	0.1564
VGG16_NCNN	1	0.9	0.95	1	1	0.85	0.95	0.1739
VGG19	0.95	0.95	0.9	1	0.8	0.85	0.9083	0.2442
VGG19_NCNN	1	0.95	1	1	0.8	0.95	0.95	0.16
ResNet50	0.9	1	0.8	0.9	1	0.95	0.925	0.2253
ResNet50_NCNN	0.95	1	0.9	1	1	0.95	**0.9667**	**0.0703**

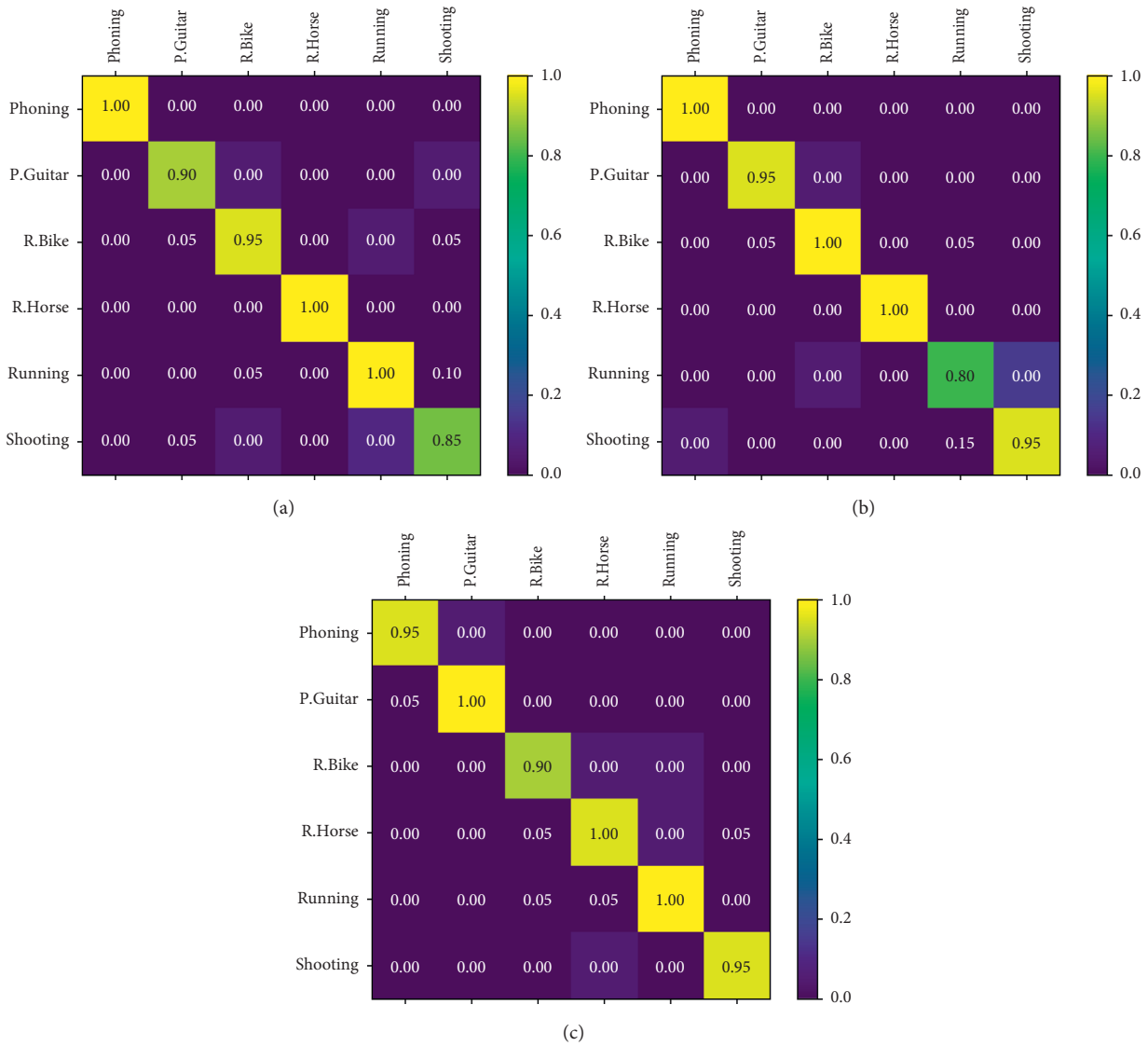


Figure 7: Confusion matrix in Li's action dataset. The i-th row represents the prediction probability when the i-th type of actions is given. (a) VGG16_NCNN. (b) VGG19_NCNN. (c) ResNet50_NCNN.

Table 2: Results for NCNN in the Willow action dataset.

Algorithm	Sensitivity for each class							Overall	
	Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc	Loss
SURF	0.03	0.01	0.01	0.01	0.02	0.02	0.01	0.1908	NA
BOF	0.55	0.43	0.42	0.48	0.40	0.26	0.13	0.3735	NA
PBOF	0.48	0.47	0.38	0.43	0.4	0.36	0.21	0.3795	NA
GIST	0.55	0.23	0.41	0.39	0.28	0.28	0.32	0.3434	NA
RF	0.28	0.39	0.35	0.44	0.36	0.28	0.07	0.3153	NA
GBM	0.45	0.28	0.33	0.41	0.26	0.40	0.17	0.3193	NA
Voting	0.52	0.33	0.39	0.38	0.30	0.31	0.21	0.3394	NA
VGG16	0.76	0.69	0.71	0.94	0.84	0.6	0.07	0.6486	1.1642
VGG16_NCNN	0.79	0.52	0.68	0.92	0.9	0.66	0.3	**0.6647**	1.0545
VGG19	0.9	0.41	0.83	0.93	0.86	0.47	0.32	**0.6647**	1.0172
VGG19_NCNN	0.83	0.51	0.72	0.86	0.88	0.67	0.23	0.6506	0.9145
ResNet50	0.93	0.39	0.7	0.89	0.78	0.41	0.51	0.6466	0.9598
ResNet50_NCNN	0.79	0.24	0.75	0.92	0.82	0.48	0.55	0.6506	**0.9019**

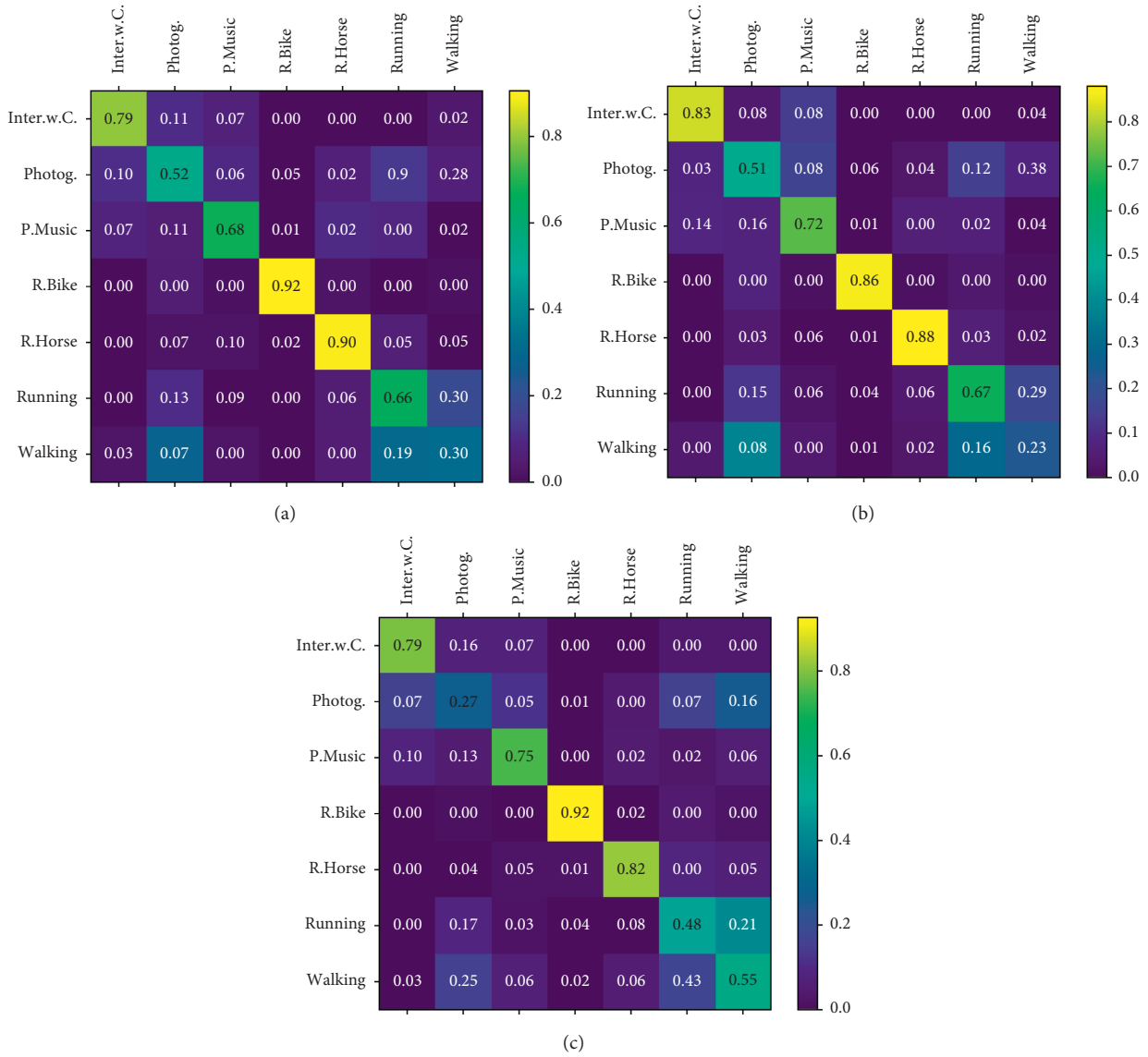


Figure 8: Confusion matrix in the Willow action dataset. The i-th row depicts the prediction probability when the i-th type of actions is provided. (a) VGG16_NCNN. (b) VGG19_NCNN. (c) ResNet50_NCNN.

Table 3: Results for NCNN in the 1.5x cropped Willow action dataset.

Algorithm	Sensitivity for each class							Overall	
	Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc	Loss
SURF	0.03	0.01	0.01	0.01	0.02	0.01	0.01	0.1406	NA
BOF	0.46	0.32	0.44	0.44	0.46	0.40	0.44	0.4234	NA
PBOF	0.49	0.34	0.47	0.48	0.58	0.47	0.33	0.4392	NA
GIST	0.59	0.36	0.31	0.45	0.49	0.43	0.30	0.3934	NA
RF	0.59	0.32	0.27	0.45	0.39	0.35	0.30	0.3618	NA
GBM	0.46	0.36	0.30	0.32	0.39	0.37	0.25	0.3270	NA
Voting	0.69	0.38	0.32	0.42	0.46	0.31	0.30	0.3791	NA
VGG16	0.62	0.19	0.73	0.88	0.46	0.59	0.41	0.5877	1.2222
VGG16_NCNN	0.74	0.53	0.76	0.82	0.6	0.56	0.33	0.6209	1.1855
VGG19	0.72	0.36	0.7	0.81	0.67	0.7	0.38	0.6209	1.1917
VGG19_NCNN	0.69	0.31	0.79	0.82	0.75	0.64	0.36	0.6272	1.0313
ResNet50	0.87	0.22	0.52	0.81	0.95	0.48	0.5	0.5987	1.1399
ResNet50_NCNN	0.85	0.16	0.67	0.87	0.77	0.23	0.74	**0.6288**	**0.9666**

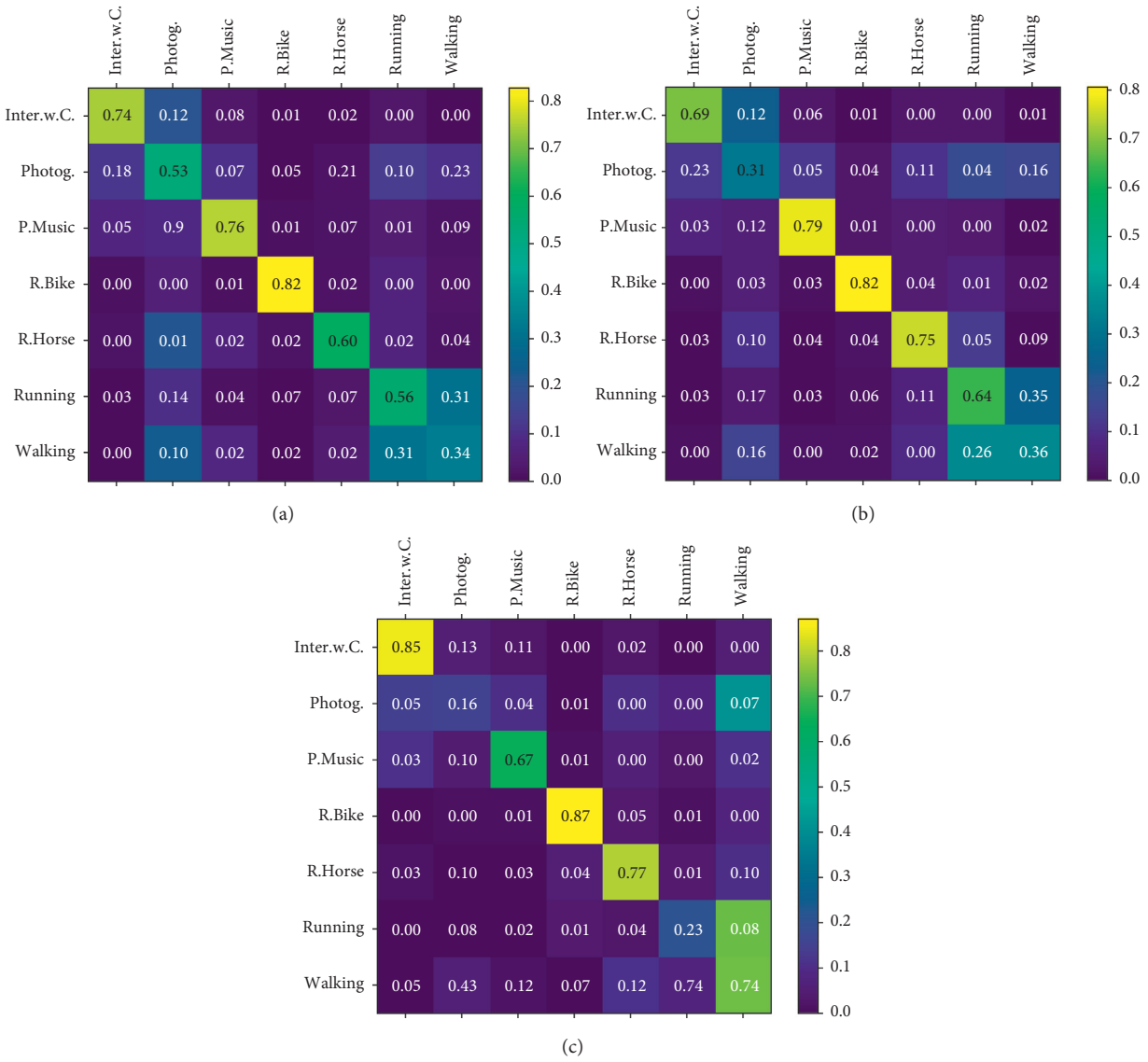


Figure 9: Confusion matrix in the 1.5x cropped Willow action dataset. The i-th row depicts the prediction probability when the i-th type of actions is provided. (a) VGG16_NCNN. (b) VGG19_NCNN. (c) ResNet50_NCNN.

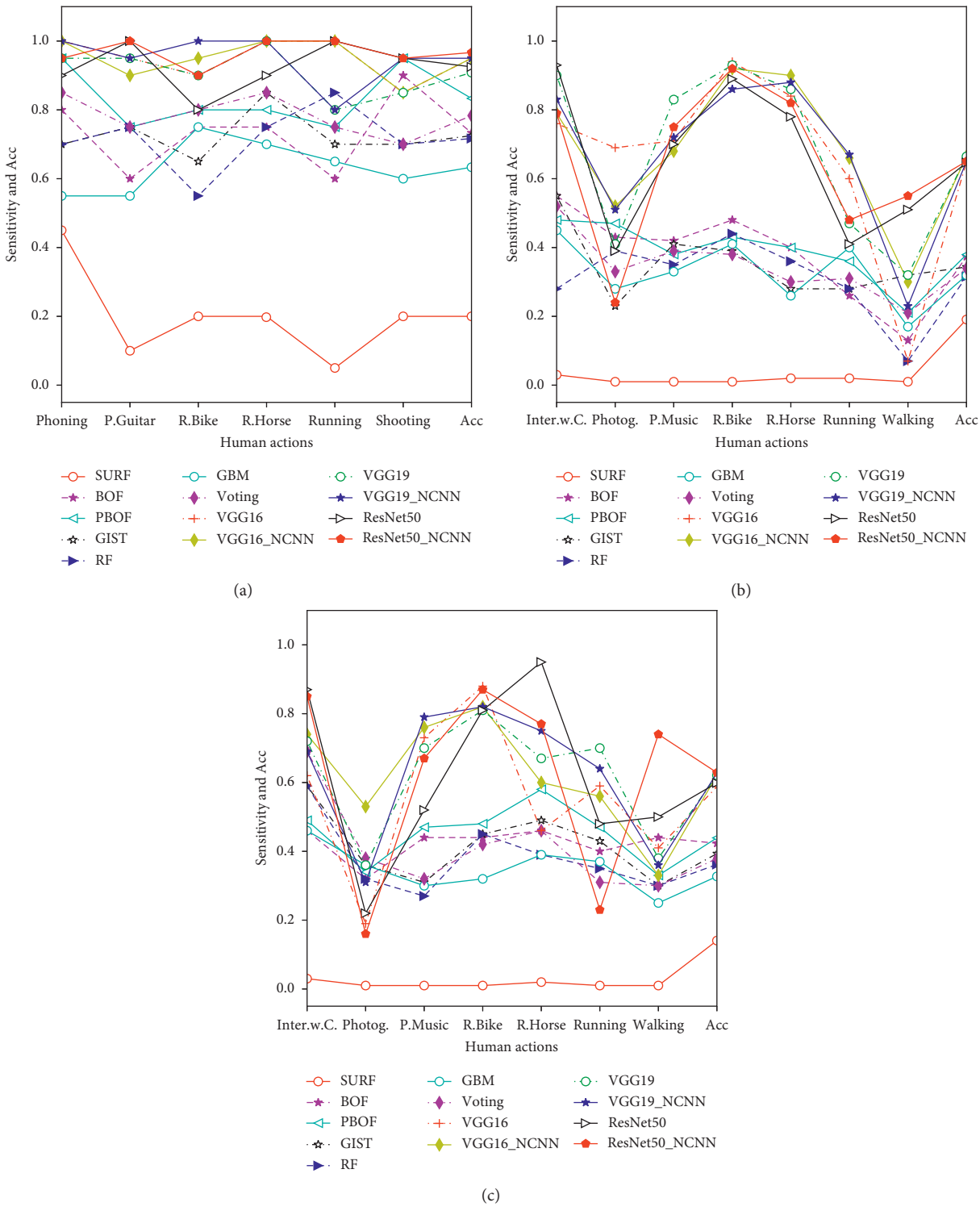


Figure 10: The classification performance of thirteen algorithms in the three datasets: results for NCNN in (a) Li's action datasets, (b) Willow's action datasets, and (c) 1.5x cropped Willow's action datasets.

Table 4: Results for DELWO in Li's action dataset.

Algorithm	Sensitivity for each class						Overall	
	Phoning	P.Guitar	R.Bike	R.Horse	Running	Shooting	Acc	Loss
DELWO1	1	0.95	0.95	1	1	1	**0.9833**	0.0844
DELWO2	1	1	1	1	0.90	0.95	0.9750	**0.0522**
DELWO3	1	0.95	1	1	1	0.95	**0.9833**	0.1035

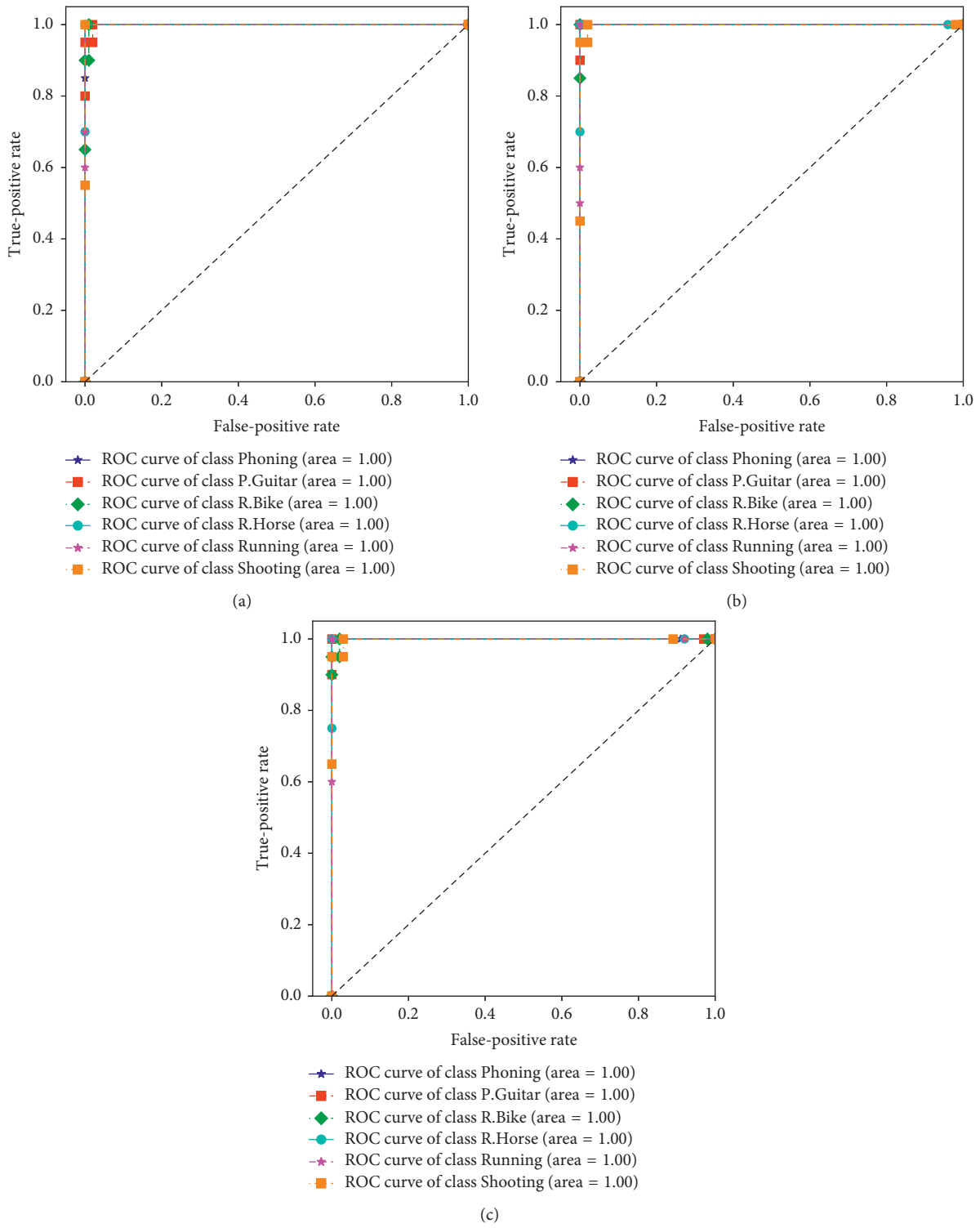


Figure 11: The ROC curves using (a) DELWO1, (b) DELWO2, and (c) DELWO3 in Li's action dataset.

Table 5: Results for DELWO in the Willow action dataset.

Algorithm	Sensitivity for each class							Overall	
	Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc	Loss
DELWO1	0.93	0.48	0.77	0.96	0.9	0.5	0.46	0.7028	**1.5195**
DELWO2	0.83	0.49	0.72	0.97	0.98	0.53	0.52	**0.7129**	1.8368
DELWO3	0.86	0.55	0.77	0.95	0.92	0.52	0.44	0.7068	1.9005

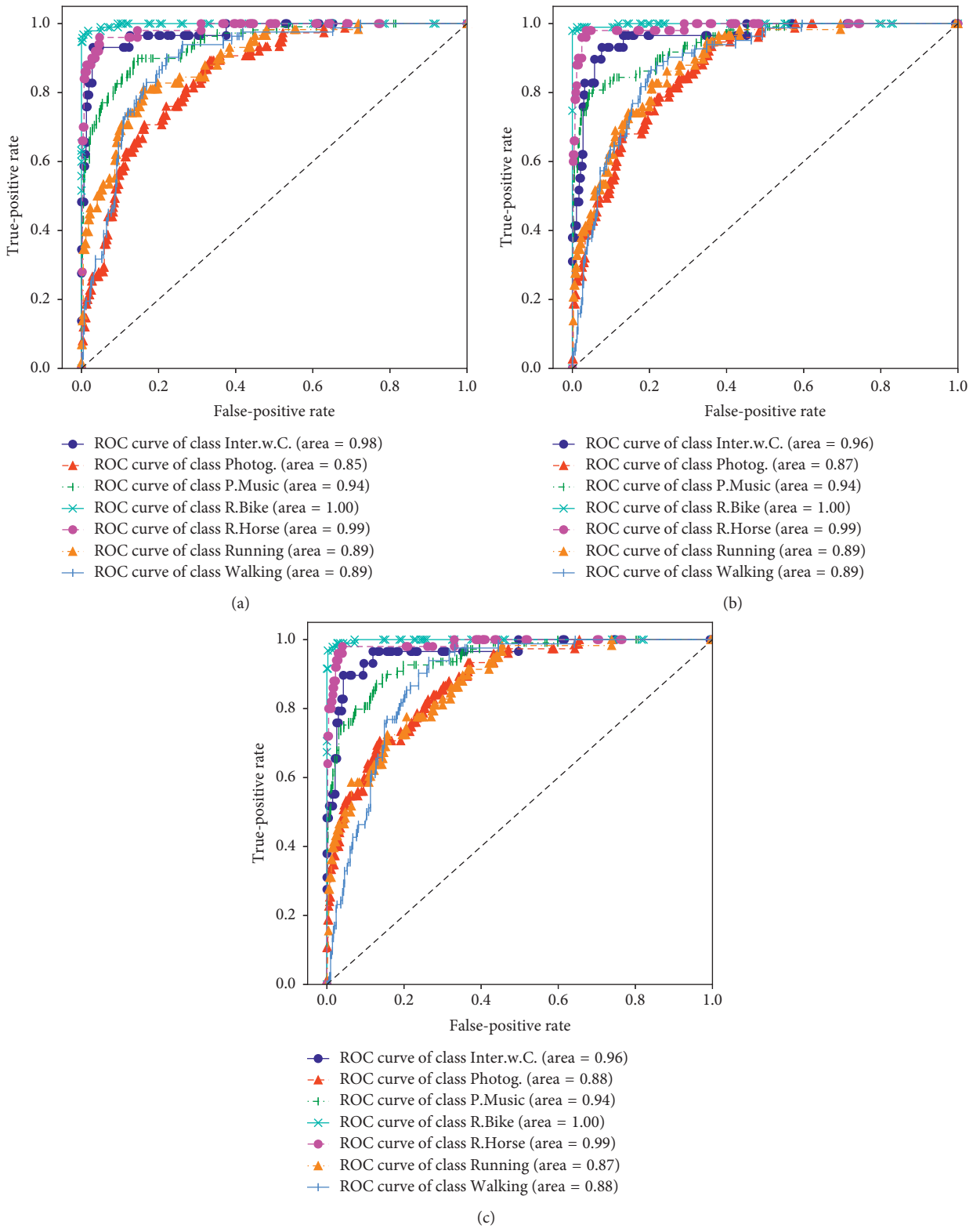


Figure 12: The ROC curves using (a) DELWO1, (b) DELWO2, and (c) DELWO3 in the Willow action dataset.

DELWO2, and DELWO3 in Li's action dataset. DELWO1 and DELWO3 obtain the best overall accuracy, and DELWO2 obtains the least loss. Compared with the performance of nonensemble models, we find all the models perform better than the best one in Table 1. Table 4 shows the specific experimental results, and Figure 11 illustrates the ROC curves using DELWO1, DELWO2, and DELWO3, which fully demonstrates the robustness of DELWO models.

Table 6: Results for DELWO in the 1.5x cropped Willow action dataset.

Algorithm	Sensitivity for each class							Overall	
	Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc	Loss
DELWO1	0.93	0.41	0.81	0.87	0.68	0.62	0.37	0.6509	1.9343
DELWO2	0.9	0.6	0.76	0.88	0.76	0.59	0.3	0.6793	**0.6793**
DELWO3	0.83	0.52	0.78	0.87	0.78	0.74	0.44	**0.6888**	1.9068

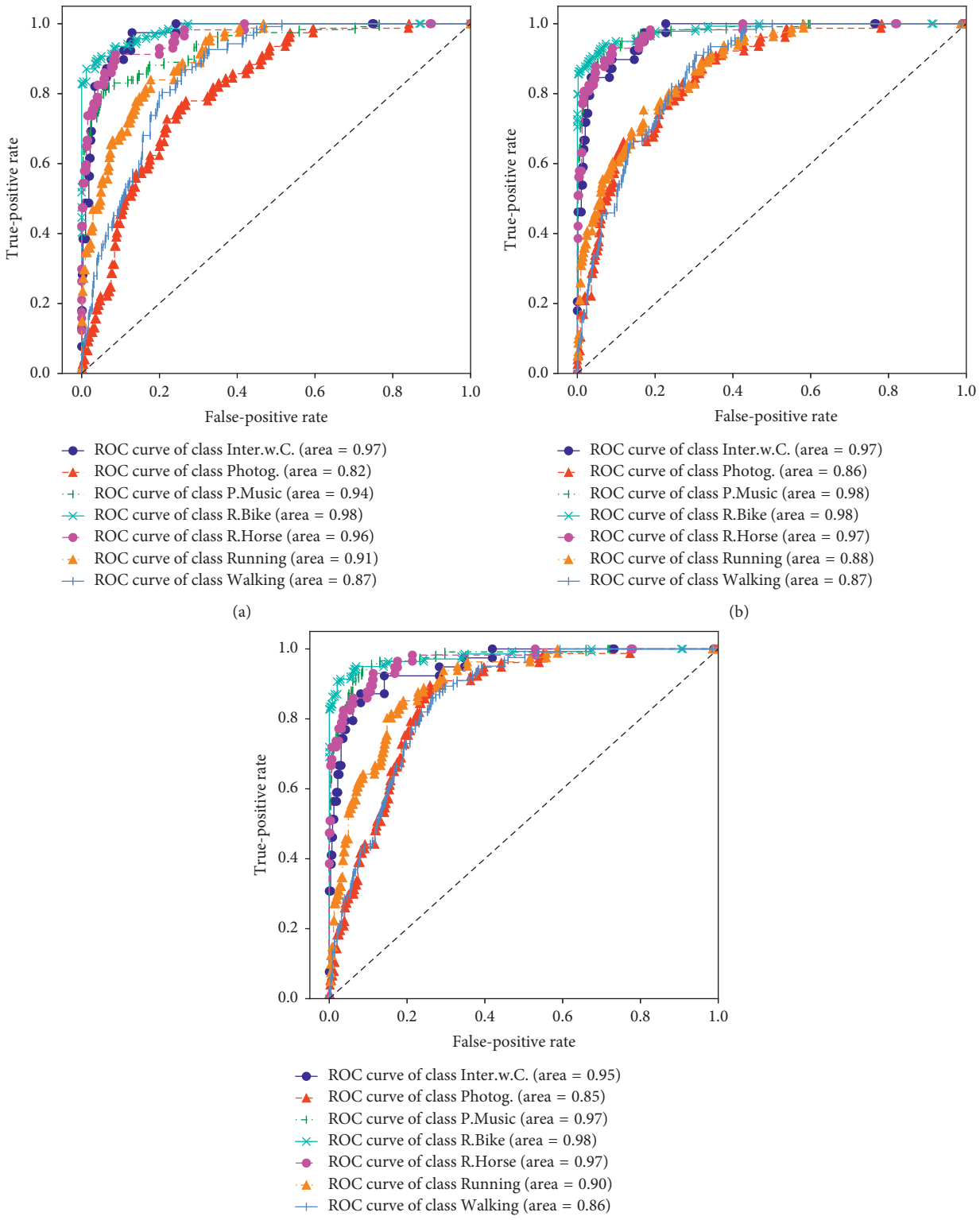


Figure 13: The ROC curves using (a) DELWO1, (b) DELWO2, and (c) DELWO3 in the 1.5x cropped Willow action dataset.

Table 7: Results for DELVS in Li's action dataset.

Model	Strategy	Sensitivity for each class						Overall
		Phoning	P.Guitar	R.Bike	R.Horse	Running	Shooting	Acc
DELVS1	Hard	1	0.95	0.9	1	1	0.95	0.9667
	Soft	1	0.95	0.9	1	1	0.95	0.9667
	Tuning	1	1	0.9	0.95	1	1	0.975
DELVS2	Hard	1	0.95	0.95	1	1	0.95	0.975
	Soft	1	0.95	0.95	1	1	1	0.9833
	Tuning	1	1	0.95	1	1	1	**0.9917**
DELVS3	Hard	1	0.95	0.9	1	1	0.95	0.9667
	Soft	1	0.95	0.9	1	1	1	0.975
	Tuning	1	1	0.95	1	1	1	**0.9917**

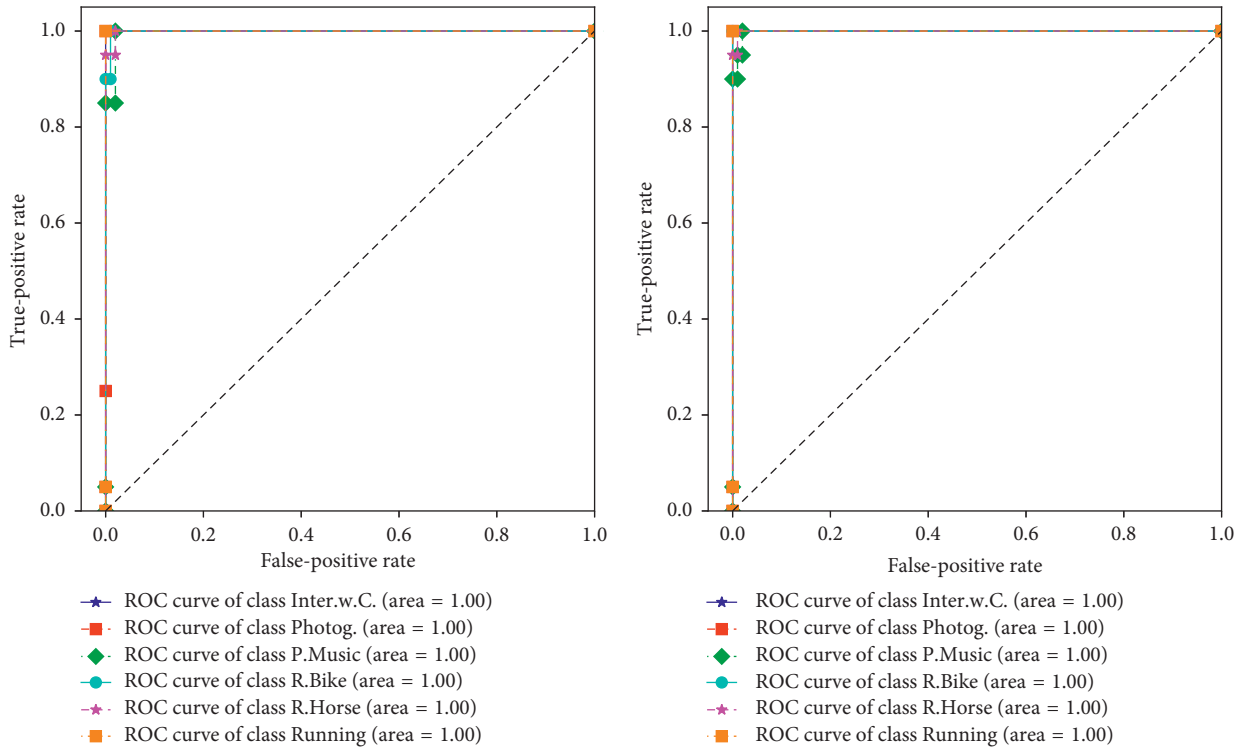
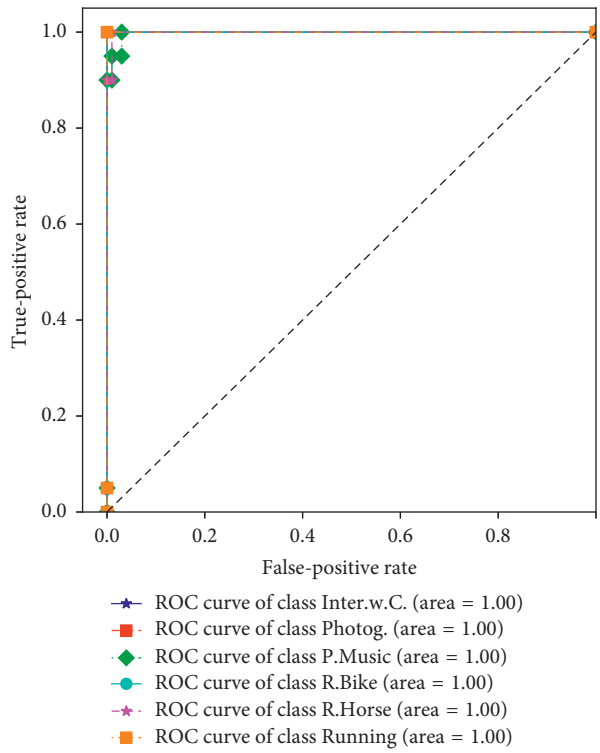


Figure 14: Continued.



(c)

Figure 14: The ROC curves using (a) DELVS1 (tuning), (b) DELVS2 (tuning), and (c) DELVS3 (tuning) in Li's action dataset.

Table 8: Results for DELVS in the Willow action dataset.

Model	Strategy	Sensitivity for each class							Overall
		Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc
DELVS1	Hard	0.9	0.57	0.75	0.97	0.88	0.52	0.22	0.6727
	Soft	0.9	0.61	0.78	0.97	0.88	0.6	0.37	0.7189
	Tuning	0.93	0.6	0.7890	0.96	0.88	0.59	0.4	0.7229
DELVS2	Hard	0.83	0.53	0.75	0.93	0.92	0.64	0.37	0.6968
	Soft	0.9	0.48	0.73	0.95	0.92	0.62	0.45	0.7048
	Tuning	0.9	0.45	0.75	0.95	0.94	0.64	0.51	0.7189
DELVS3	Hard	0.9	0.59	0.75	0.95	0.94	0.62	0.32	0.7048
	Soft	0.93	0.56	0.76	0.95	0.92	0.62	0.41	0.7189
	Tuning	0.9	0.65	0.79	0.95	0.94	0.64	0.39	**0.7369**

Table 5 shows the results of using DELWO1, DELWO2, and DELWO3 in the Willow action dataset. DELWO2 obtains the best overall accuracy, DELWO3 obtains the second-best one, and DELWO1 obtains the least loss. Compared with the performance of nonensemble models in the Willow action dataset, DELWO models have improved by almost 5%. The specific performance is shown in Table 5 and Figure 12.

Similarly, Table 6 shows the results of using DELWO1, DELWO2, and DELWO3 in the 1.5x cropped Willow action dataset. DELWO3 obtains the best overall accuracy, and DELWO2 obtains the least loss. Compared with the performance of nonensemble models in the 1.5x cropped Willow action dataset, DELWO models have improved by almost 6%. Particularly, the DELWO3 performs best when identifying the "Running" action. The detailed performance is shown in Table 6 and Figure 13.

4.2.3. Results for Deep Ensemble Learning Based on Voting Strategy. Table 7 presents the results of using DELVS1, DELVS2, and DELVS3 in Li's action dataset. Comparing Table 7 with Table 1, we can see that the performance of the DELVS model is better than that of the NCNN-based model. It is worth noting that DELVS2 (tuning) and DELVS3 (tuning) obtain better results over DELWO1 and DELWO3 in Li's action dataset. In general, the tuning weight voting method will achieve the best results among these three voting strategies. The detailed performance of DELVS models is elaborated in Table 7 and Figure 14.

Table 8 presents the results of using DELVS1, DELVS2, and DELVS3 in the Willow action dataset. Comparing Table 8 with Table 2, we can see that the performance of the DELVS model is better than that of

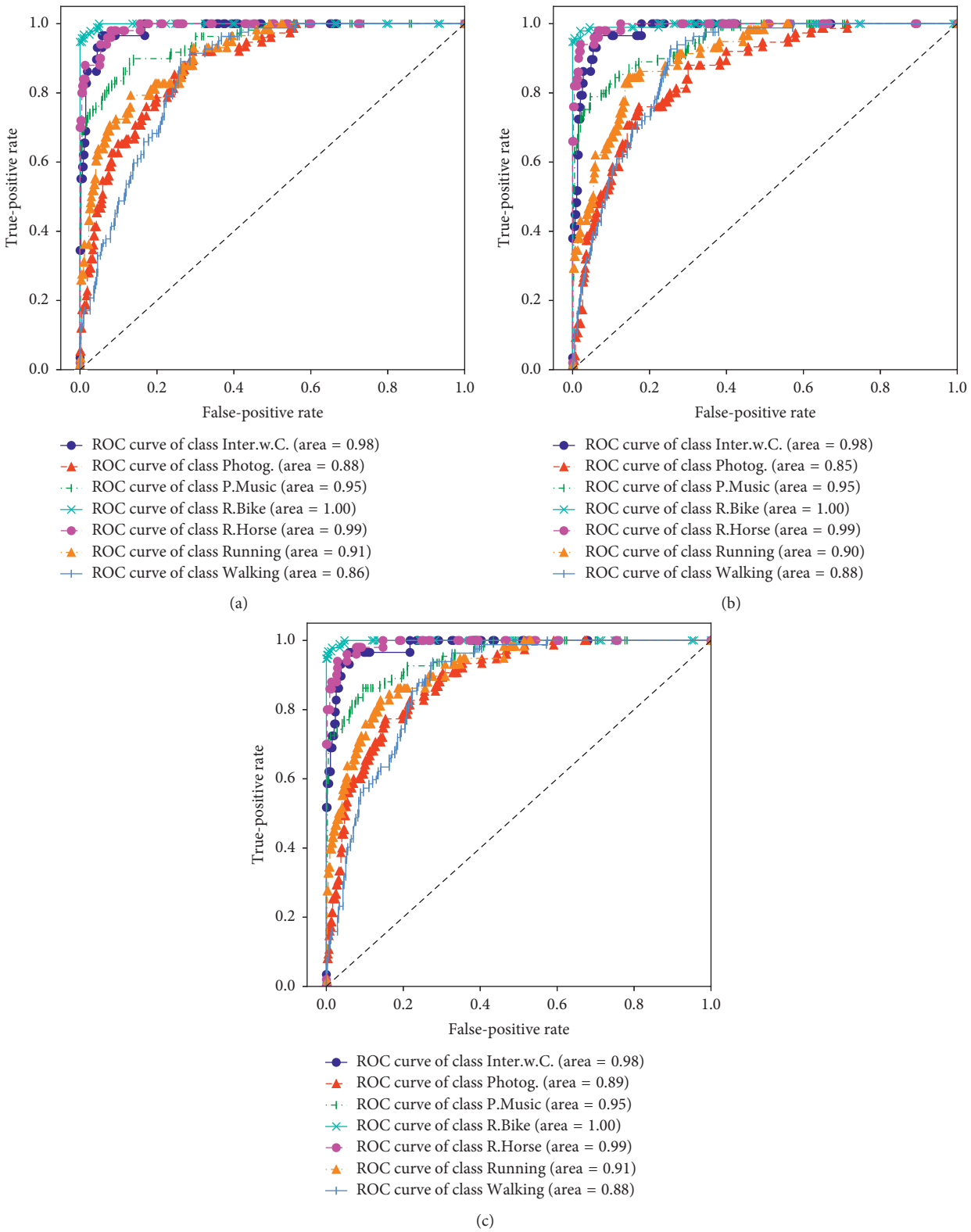


Figure 15: The ROC curves using (a) DELVS1 (tuning), (b) DELVS2 (tuning), and (c) DELVS3 (tuning) in the Willow action dataset.

the NCNN-based model. It is worth noting that DELVS3 (tuning) obtains better results in the Willow action dataset. And the tuning weight voting method has achieved best results among these three voting strategies. The detailed performance of DELVS models is elaborated in Table 8 and Figure 15.

Similarly, we can reach conclusion that the tuning weight voting method performs best among these three voting

Table 9: Results for DELVS in the 1.5x cropped Willow action dataset.

Model	Strategy	Sensitivity for each class							Overall
		Inter.W.C.	Photog.	P.Music	R.Bike	R.Horse	Running	Walking	Acc
DELVS1	Hard	0.87	0.35	0.69	0.84	0.77	0.62	0.39	0.6351
	Soft	0.82	0.27	0.72	0.86	0.75	0.65	0.43	0.6414
	Tuning	0.82	0.3	0.73	0.86	0.74	0.68	0.43	0.6461
DELVS2	Hard	0.82	0.45	0.77	0.84	0.75	0.51	0.43	0.6509
	Soft	0.82	0.40	0.79	0.84	0.79	0.54	0.47	0.6619
	Tuning	0.85	0.29	0.76	0.86	0.86	0.51	0.61	**0.6777**
DELVS3	Hard	0.85	0.4	0.74	0.85	0.81	0.58	0.4	0.6493
	Soft	0.85	0.34	0.76	0.84	0.81	0.58	0.46	0.6556
	Tuning	0.85	0.4	0.79	0.84	0.79	0.59	0.49	0.6746

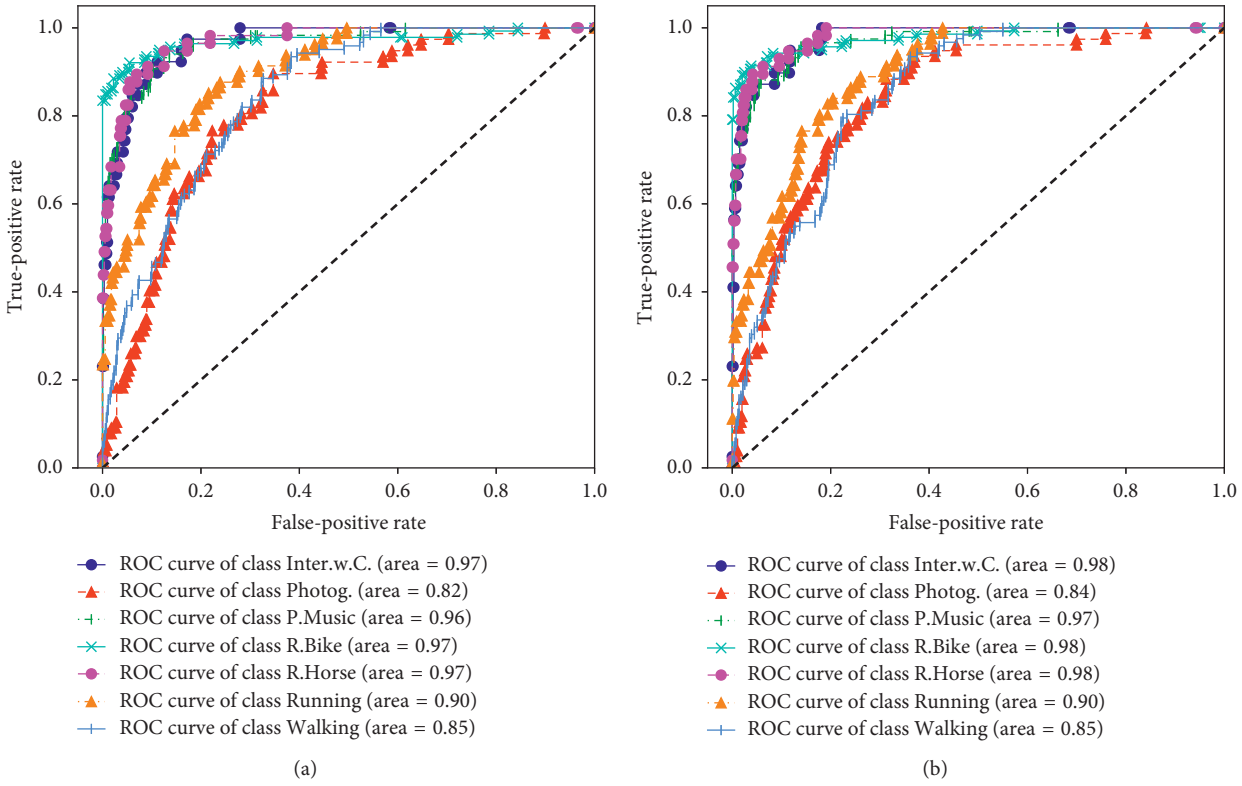
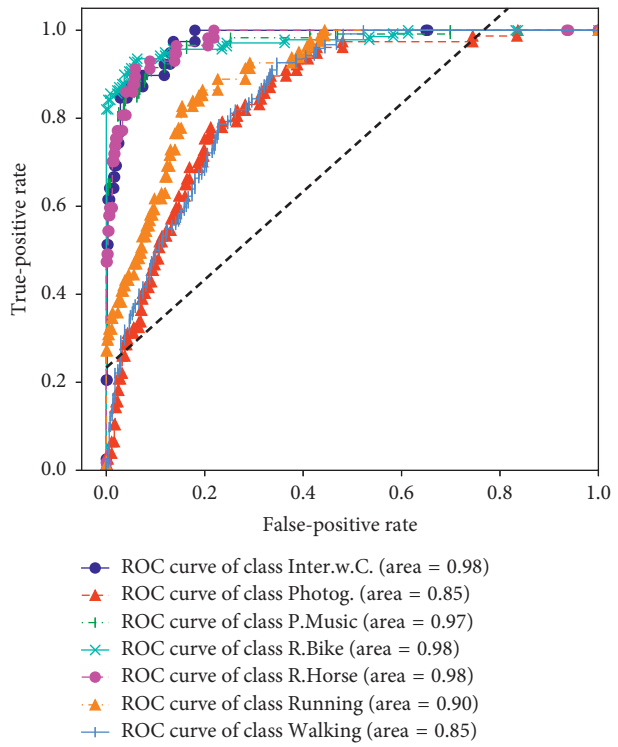


Figure 16: Continued.



(c)

Figure 16: The ROC curves using (a) DELVS1 (tuning), (b) DELVS2 (tuning), and (c) DELVS3 (tuning) in the 1.5x cropped Willow action dataset.

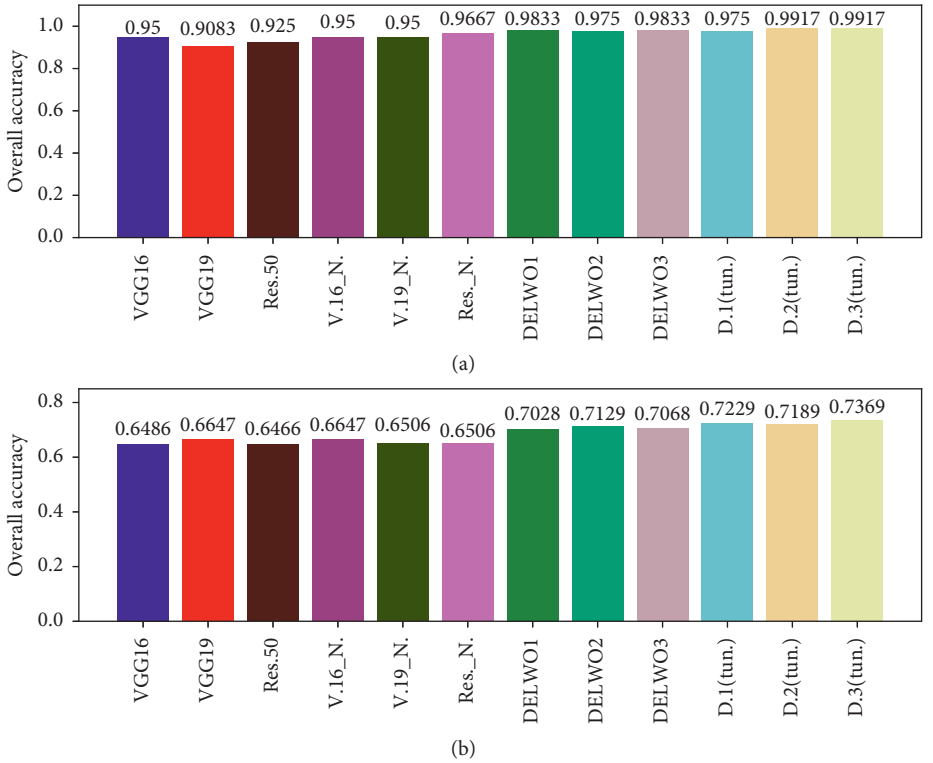


Figure 17: Continued.

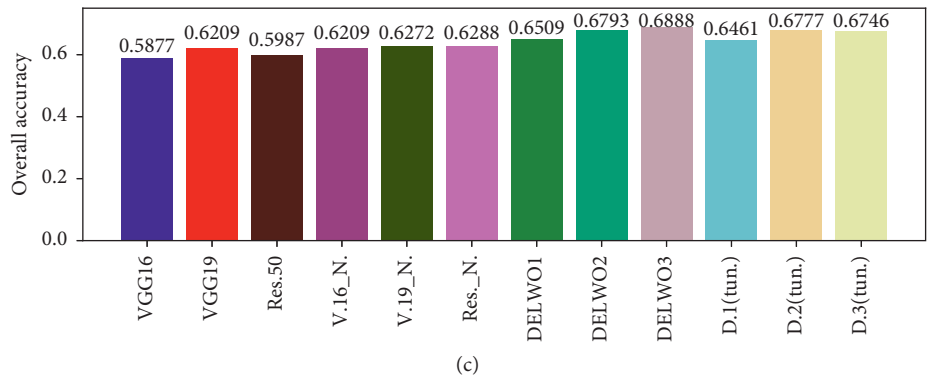


Figure 17: The overall accuracy of our proposed models compared with other models: overall accuracy in (a) Li's action dataset, (b) Willow action dataset, and (c) 1.5x cropped Willow action dataset.

Figure 18: The class activation heatmaps using different deep models.

strategies. Compared with the performance of nonensemble models in the 1.5x cropped Willow action dataset, the DELVS model has improved by almost 5%. However, the performance of the best DELVS2 (tuning) does not exceed that of the DELWO3. It shows that DELWO model is more competent in this dataset. The detailed performance of the DELVS model in the 1.5x cropped Willow action dataset is elaborated in Table 9 and Figure 16.

4.3. Experimental Analysis. From Figure 17, we can conclude that the deep ensemble models outperform the nonensemble ones. DELVS models obtain the best results in Li's action dataset and Willow action dataset, while DELWO models obtain the best results in the 1.5x cropped Willow action dataset. This can fully explain that the deep ensemble models can better discover more aspects of the "truth" and show robustness when it is faced with interference.

Comparing the experimental results in the 1.5x cropped Willow action dataset, we can draw another conclusion that all the deep models perform better in terms of the overall accuracy in the Willow action dataset. Therefore, we can speculate that the "background" information of each action can provide useful signals and cues for classifying corresponding actions. For example, the "InteractingWithComputer" actions usually occur indoors, while the "RidingBike," "RidingHorse," etc. often occur outdoors. The "background" information is usually linked with specific actions, so it is valuable to incorporate the "background" information when classifying corresponding human actions.

Figure 18 shows the detailed class activation heatmaps using Grad-CAM [45] for different deep models among three "RidingHorse" actions. The NCNN-based models will detect more response area than the baseline ones to some degree. Although DELWO models have found less response area, they retain the most core part and are more compact ones. We speculate that this may be the reason why the DELWO models show greater robustness for classifying these actions.

5. Conclusions

In this research, we propose the deep ensemble learning approaches to automatically perform human action recognition in still images. Human actions such as "Phoning," "RidingHorse," and "Running" require static cue-based approaches due to the nature of these actions. Recognizing human actions in still images is complementary to video-based methods. How to mitigate overfitting has always been one of the most challenging tasks in computer vision and machine learning. It becomes more intractable when a deep learning model needs to be trained in small datasets. Therefore, how to well mitigate overfitting when training our model is an important issue.

To solve the above issues, first, the weights of the convolution layer module of our models are initialized by pretrained models in terms of transfer learning. In addition, we adopt the data augmentation technique to generate more training data to further mitigate overfitting. Second, the GAP trick can greatly shrink the number of trainable parameters. Therefore, our models are lighter and generalize well to unseen data. Furthermore, it is feasible to train our novel model on a single PC with CPU benefiting from the end-to-end structure. Moreover, the nonsequential network topology facilitates the NCNN-based model to separately learn the spatial- and channel-wise features for parallel branches. The DELWO model, a generalized nonsequential network topology, can fuse deep features among multiple models automatically from data. The DELVS model can pool together different classifiers to produce a better prediction.

Our experimental results reveal that the "background" information may provide helpful signals and cues for classifying human actions. Incorporating the action and background information will be part of our follow-up work. The nonsequential network topology possesses powerful advantages over traditional sequential topologies, and thus, further developing a nonsequential model with separable convolution layers and multiple inputs will be another line of our follow-up research. For example, we can utilize the action information and background information as two independent inputs to jointly learn a nonsequential model. That is to say, the nonsequential model can be trained by utilizing multiple modalities of inputs concurrently, and this idea will be the focus of our follow-up research.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the Doctoral Scientific Research Foundation of Jiangxi University of Science and Technology (Grant no. jxxjbs19029), Science and Technology Developing Project of Jilin Province, China (Grant no. 20150204007GX), Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, Science and Technology Development Plan Project of Jilin Province (Grant no. 20180520017JH), and Science and Technology Project of the Jilin Provincial Education Department (Grant no. JJKH20170107KJ).

References

[1] V. Delaitre, I. Laptev, and J. Sivic, "Recognizing human actions in still images: a study of bag-of-features and part-based representations," in *Proceedings of the BMVC 2010-21st British Machine Vision Conference*, Aberystwyth, UK, August 2010.

[2] G. Yao, T. Lei, and J. Zhong, "A review of convolutional-neural-network-based action recognition," *Pattern Recognition Letters*, vol. 118, pp. 14–22, 2019.

[3] W. Xu, Z. Miao, J. Yu, and Q. Ji, "Action recognition and localization with spatial and temporal contexts," *Neurocomputing*, vol. 333, pp. 351–363, 2019.

[4] M. Majd and R. Safabakhsh, "Correlational convolutional LSTM for human action recognition," *Neurocomputing*, 2019.

[5] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1933–1941, Las Vegas, NV, USA, July 2016.

[6] H. Bilen, B. Fernando, E. Gavves, and A. Vedaldi, "Action recognition with dynamic image networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2799–2813, 2018.

[7] P. Li, J. Ma, and S. Gao, "Actions in still web images: visualization, detection and retrieval," in *Proceedings of the International Conference on Web-Age Information Management*, pp. 302–313, Wuhan, China, September 2011.

[8] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[9] Y. Zhang, L. Cheng, J. Wu, J. Cai, M. N. Do, and J. Lu, "Action recognition in still images with minimum annotation efforts," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5479–5490, 2016.

[10] F. S. Khan, R. Muhammad Anwer, J. Van De Weijer, A. D. Bagdanov, A. M. Lopez, and M. Felsberg, "Coloring action recognition in still images," *International Journal of Computer Vision*, vol. 105, no. 3, pp. 205–221, 2013.

[11] G. Guo and A. Lai, "A survey on still image based human action recognition," *Pattern Recognition*, vol. 47, no. 10, pp. 3343–3361, 2014.

[12] S. Abidi, M. Piccardi, and M. A. Williams, "Action recognition in still images by latent superpixel classification," 2015, https://arxiv.org/abs/1507.08363.

[13] Z. Liang, X. Wang, R. Huang, and L. Lin, "An expressive deep model for human action parsing from a single image," in *Proceedings of the 2014 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, IEEE, Chengdu, China, July 2014.

[14] T. Qi, Y. Xu, Y. Quan, Y. Wang, and H. Ling, "Image-based action recognition using hint-enhanced deep neural networks," *Neurocomputing*, vol. 267, pp. 475–488, 2017.

[15] J. Kong, B. Zan, and M. Jiang, "Human action recognition using depth motion maps pyramid and discriminative collaborative representation classifier," *Journal of Electronic Imaging*, vol. 27, no. 3, Article ID 033027, 2018.

[16] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: using spatial and functional compatibility for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1775–1789, 2009.

[17] G. Zhang, S. Jia, X. Zhang, and X. Li, "Saliency-based foreground trajectory extraction using multiscale hybrid masks for action recognition," *Journal of Electronic Imaging*, vol. 27, no. 5, Article ID 053049, 2018.

[18] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[19] B. Ko, J. Hong, and J.-Y. Nam, "Human action recognition in still images using action poselets and a two-layer classification model," *Journal of Visual Languages & Computing*, vol. 28, pp. 163–175, 2015.

[20] L. Cai, X. Liu, F. Chen, and M. Xiang, "Robust human action recognition based on depth motion maps and improved convolutional neural network," *Journal of Electronic Imaging*, vol. 27, no. 5, Article ID 051218, 2018.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[22] Y. LeCun, B. E. Boser, J. S. Denker et al., "Handwritten digit recognition with a back-propagation network," *Advances in Neural Information Processing Systems*, vol. 2 pp. 396–404, 1990.

[23] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[24] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, https://arxiv.org/abs/1312.4400.

[25] A. M. Nickfarjam and H. Ebrahimpour-Komleh, "Shape-based human action recognition using multi-input topology of deep belief networks," in *Proceedings of the 2017 9th International Conference on Information and Knowledge Technology (IKT)*, pp. 1–4, IEEE, Tehran, Iran, October 2017.

[26] O. Oktay, W. Bai, M. Lee et al., "Multi-input cardiac image super-resolution using convolutional neural networks," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 246–254, Athens, Greece, October 2016.

[27] Y. Sun, L. Zhu, G. Wang, and F. Zhao, "Multi-input convolutional neural network for flower grading," *Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 9240407, 8 pages, 2017.

[28] Y. Fujita, R. Takashima, T. Homma, and M. Togami, "Data augmentation using multi-input multi-output source separation for deep neural network based acoustic modeling," in *Proceedings of the Interspeech*, pp. 3818–3822, San Francisco, CA, USA, September 2016.

[29] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: a survey," *Information Fusion*, vol. 37, pp. 132–156, 2017.

[30] X.-L. Zhang and D. Wang, "A deep ensemble learning method for monaural speech separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 967–977, 2016.

[31] F. Chollet and J. J. Allaire, "Advanced deep-learning best practices," in *Deep learning with R*, vol. 218–249, Manning Publications Co., Shelter Island, NY, USA, 2018.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[33] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Honolulu, HI, USA, July 2017.

[34] A. G. Howard, M. Zhu, B. Chen et al., "Mobilenets: efficient convolutional neural networks for mobile vision applications," 2017, https://arxiv.org/abs/1704.04861.

[35] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[36] J. Sivic and A. Zisserman, "Video google: a text retrieval approach to object matching in videos,"in *Proceedings NinthIEEE International Conference on Computer Vision*, p. 1470, IEEE, October 2003.

[37] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition—Volume 2 (CVPR'06)*, pp. 2169–2178, IEEE, Washington, DC, USA, 2006.

[38] R. Ye and P. N. Suganthan, "Empirical comparison of bagging-based ensemble classifiers," in *Proceedings of the 2012 15th International Conference on Information Fusion*, pp. 917–924, IEEE, Singapore, July 2012.

[39] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[40] H. Drucker, C. Cortes, L. D. Jackel, Y. LeCun, and V. Vapnik, "Boosting and other ensemble methods," *Neural Computation*, vol. 6, no. 6, pp. 1289–1301, 1994.

[41] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in Neurorobotics*, vol. 7, p. 21, 2013.

[42] R. Polikar, "Ensemble learning," in *Ensemble Machine Learning*, pp. 1–34, Springer, Boston, MA, USA, 2012.

[43] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[44] S. Shah, K. Khatri, P. Mhasakar, R. Nagar, and S. Raman, "Unsupervised GIST based clustering for object localization," in *Proceedings of the 2019 National Conference on Communications (NCC)*, pp. 1–6, IEEE, Atlanta, GA, USA, February 2019.

[45] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626, Venice, Italy, October 2017.